ABSTRACT

UInterestManager: A Generic Framework to Model User Interests

By Majid Darabi

April, 2013

Director of Thesis: Dr. M. H. Nassehzadeh Tabrizi

Major Department: Computer Science

With the proliferation of documents on the Internet, it has become increasingly difficult for users to search and retrieve the information they want: information that is relevant and interesting. To address this problem, researchers have adopted personalized services and filtering systems that rely on user preferences. In this thesis, we present a novel method to model user interests automatically and generate a ranked list of preferences for each user in real time.

The proposed framework here consists of three main steps to automatically model user preferences and generate a ranked list of the user's interests in real time. 1) Through the use of WordNet, a list of keywords which represent explicit interests is extracted from a text document and then implicit user preferences are determined by employing domain ontology. 2) A hierarchical data structure for each user stores interests and is updated for each new document read by that user. Additionally, there is a queue of recently accessed interests for each user that maintains these preferences based upon arrival time. 3) The final step includes generating a ranked list of top K interests for each user in real time. This ranking system employs the user model to create a combined list of both short-term and long-term interests. The results show significant improvement over some widely used methods.

UInterestManager: A Generic Framework to Model User Interests

A Thesis

Presented to the Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science

by

Majid Darabi

April, 2013

UInterestManager: A Generic Framework to Model User Interests

By

Majid Darabi

APPROVED BY:

DIRECTOR OF THESIS:_____
M. H. Nassehzadeh Tabrizi, PhD

COMMITTEE MEMBER:_____
Karl Abrahamson, PhD

COMMITTEE MEMBER:_____
Qin Ding, PhD

CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE:
_____
Karl Abrahamson, PhD

DEAN OF THE GRADUATE SCHOOL:
_____
Paul J. Gemperline, PhD

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

People's interests allow them to focus on interesting and related information and to filter unrelated data. Extracting implicit user interests from text documents and modeling users' short-term and long-term preferences are core components for providing personalized information. This thesis will consider the problem of modeling user interests and ranking preferences in real time.

1.1 Modeling User Interests

As a result of the exponential growth of the Internet, the quantity and diversity of published content have increased. This poses a significant challenge to users' ability to quickly access the information they need. In order to solve this problem, personalized services and recommendation systems have been employed in order to supply relevant information of interest to users. Thus, modeling user interests has played a crucial role in personalization and information filtering applications, such as recommender systems, that attempt to meet user needs [1]. Moreover, accessing individual user interests helps web services and user-centric applications provide better, personalized information for the user.

Generally, the process for modeling user interests requires two steps: 1) recognizing user interests; and 2) creating a model of those interests [4]. Systems that recognize interests may employ implicit indicators, explicit indicators, or a combination of both. With explicit indicators, an application requests that users directly enter their interests or provide feedback through ratings [2]; however, entering lists of interests or rating documents is a tedious and time-

consuming process. In addition, since user interests are often diverse and broad, many users are unable to provide a complete and detailed list of their interests. Also, studies show people generally only provide feedback for small portions of documents that they read [5].

Analyzing user interactions with documents and the context of the documents helps to extract user interests automatically [3]. In other words, implicit indicators focus on browsing behaviors (such as duration) and browsing contents in order to recognize implicit user interests [6, 7, 8]. User browsing behaviors offer value in modeling user interests, which may provide a great contribution to user preferences [10, 11].

The degrees of influence for capturing user interests related to interacted documents are not the same. There are two types of documents that contain user interests: generated documents and read documents. Generated documents are a valuable source of information for inferring implicit user interests, as they are created completely based upon user thoughts and preferences. Conversely, a user may only be interested in part of a read document or it may be that the document is only close but does not necessarily include the user interests. Therefore, a generated document or a text by a user provides not only more information, but more reliable information about the user's preferences, as opposed to a read document [3].

User preference contains some elements that must be considered in order to create an accurate and effective model. Naturally, people's interests are not static and change over time. Their job, life, and many other factors may influence or change their preferences [9]. For example, if a programmer changes careers and becomes a salesman, those interests will naturally shift with this change. Therefore, user preferences may be divided into two categories based upon interest importance over time: 1) short-term interests; and 2) long-term interests [9]. Short-

term interests refer to user preferences that receive attention from the user for a short period and change over time. Unlike short-term interests, long-term preferences are not subject to frequent changes over time. For instance, a machine learning expert requires some knowledge about Java programming for a project. In this example, "Machine Learning" is a long-term interest and most likely will not change over time while "Java Programming" is a short-time preference and the user may not be interested in it after finishing the project.

1.2 Thesis Contribution

In this thesis, we propose a framework that captures implicit user interests, creates a model user profile, and creates a ranked list of user interests in real time. The framework receives a document with the reading time and user information. Then it extracts implicit interests and updates the user model. Finally, the framework generates a ranked list of user preferences in real time, considering the incoming query.

In order to recognize implicit interests from a text document, WordNet [31] and domain ontologies are employed. Semantic reasoning with domain ontology is utilized to infer implicit preferences. User interests are characterized as domain concepts, and implicit interests are obtained by inference with superclass and subclass relations. By applying this approach, the context of the explicit interests may be acquired. The reading time of a given text document is used to calculate the degree of interest in the content. The extracted preferences and the degree of interest in the document will be used to update the user profile. The proposed model is able to maintain changes in interest over time and can manage both long-term and short-term interests.

The framework produces a ranked list of interests for each user in real time. The framework returns the top N preferences, including long-term and short-term interests. This list may be used by different applications that require user interests to provide personalized information. For example, user preferences are used by search engines to personalize search results and recommendation systems to recommend those personalized items. Therefore, the contributions of this thesis are as follows:

I. Using WordNet and domain ontology to automatically extract implicit interests from text documents.

II. Modeling the long-term and short-term preferences.

III. Generate a ranked list of interests for each user in real time.

1.3 Thesis Overview

The remainder of this thesis is as follows:

- Chapter 2 will discuss different studies related to extracting user interests and modeling user preferences.
- Chapter 3 will outline the workflow of the system and general overview of components of the UInterestManager.
- Chapter 4 will explain the model and its details implementation.
- Chapter 5 will present the results and evaluations with real data.

- Chapter 6 will summarize the main points of the thesis and will show possible new directions for future works.

CHAPTER 2: RELATED WORK

Capturing a user's interests is a major challenge in providing effective personalized information. Since users are reluctant to specify their preferences, asking users to provide their interests is not an effective method. Therefore, techniques that extract implicit information about users' preferences have become popular among researchers [12, 13].

Traditional methods, e.g., [14, 15] use keywords with associated weights in order to model user interests. However, the vagueness of words is the major shortcoming in modeling user interests represented by keywords and terms. As a result of the ambiguity of keywords, this type of modeling user profile may not represent user interests accurately, resulting in poor performance. To overcome the drawbacks of the Bag of Words model, ontology based user models were developed [16, 17, 18, 9]. Ontology can demonstrate the relationship between words, support semantic reasoning and represent a clear conceptual definition of the resources.

Pretschner and Gauch [18] are the pioneers in modeling user interests by employing ontology and by creating personalized document access. They used domain ontology to organize documents, and by analyzing a user's browsing history, could model the user interest. This study deemed user profiles to be static and did not consider the change of interests over time. Li et al [9], proposed independent models to model both long-term and short-term preferences. They used Google Directory to build a taxonomic hierarchy for the long-term model, and visited page history window for modeling short-term interests. This approach was deemed to be an ontology-based user profile, since the model is structured hierarchically. A major problem with this work, however, is that they only consider the URL of a page, which is unable to capture user preference accurately and results in poor performance over time.

Ma et al [3] presented a user interests model using ontology based upon fusion and semantic reasoning. The proposed method integrates and analyzes user-generated contents in different web platforms, and utilizes semantic reasoning and domain ontology in order to model user interests. The system represents dynamic interests that refer to the information with created or update time, and static preferences that present information with no tag time, such as professional interests. Unfortunately, it is not known whether the system can model long-term and short-term interests. Additionally, they use information fusion strategies to produce a ranked list of user preferences without considering time and short-term interests that result in inaccurate generated ranked lists.

Sieg et al [17] proposed ontology-based user profiles for personalized search. The system utilizes Open Directory Project (ODP) taxonomy to build the web topic ontology. They use the specific configuration of Spreading Activation Theory to maintain interest scores within a user profile. In this research, a list of interested documents for a user is provided to the system, which starts incrementally calculating interest scores. Finally, a re-ranking module is developed in order to tailor the search results to the user. This system is unable to provide a good performance in real time, as it uses batch processing to update user profiles. Moreover, they do not consider attention to an interest during a period of time, which results in inaccurate updates of short-term interests.

Jiang and Tan [16] presented an ontology based user model, called user ontology to capture the users' interests in a working domain to provide personalized web search. The user ontology differs from existing approaches by using concepts, taxonomic relations, and non-taxonomic relations in a given domain ontology. The authors introduced a set of statistical methods for learning a user model and employed Spreading Activation Theory to process

information in the user ontology. Google Directory and ACM digital library were used by the authors to evaluate the model. However the user ontology does not separate the long-term and short-term interests therefore it cannot provide the accurate list of interests in real time. Moreover, the model does not consider the sequence of interests with their time stamps that result in poor performance over time.

CHAPTER 3: OUR APPROACH

In this chapter the core ideas behind our approach will be explained without revealing too many implementation details.

3.1 UInterestManager Workflow

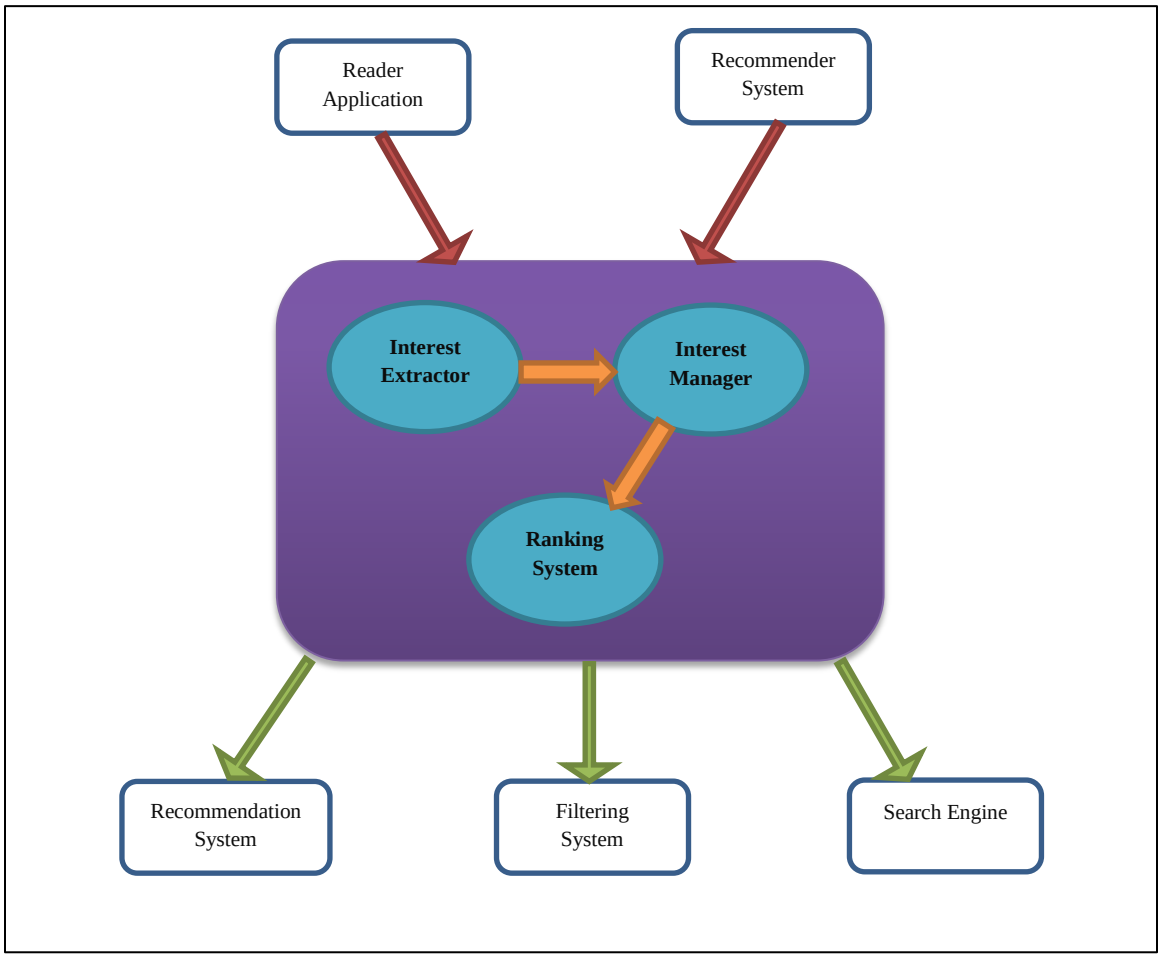

Figure 3.1 UInterestManager workflow diagram

As shown in Figure 3.1, the UInterestManager receives inputs, updates the user model, and generates outputs based upon a given query. The input is a triple which contains the user's id, a text document read by the user, and an interestingness factor that is determined by the sender application. Then, the UInterestManager manipulates input data for the user and updates preferences using the user model. The system generates a ranked list of top K user interests in real time. Finally, the UInterestManager receives a user's id and employs the user information and preferences to produce a ranked list of interests that provides the current user preferences. The output of the system is a ranked list of top K user interests in real time.

UInterestManager could be used by applications that provide customized information for users. For example, reader applications or recommendation systems can send user's information to UInterestManager and obtain the preferences of the user to personalize information or to recommend documents based upon user interests. Additionally, other information provider services such as search engines or filtering systems can also utilize the UInterestManager in order to create tailored and personalized information.

3.2. Extracting Explicit and Implicit Interests

The first step in capturing user interests from text documents is through extracting explicit and implicit interests from a given document. The Interest Extractor is a sub system of UInterestManager that is responsible for extracting preferences. Given a text document, Interest Extractor finds all frequently used words, and then by using WordNet, determines all high frequency terms and disambiguates word senses. Finally, it sends a list of extracted keywords and terms to the Interest Manager.

WordNet is a lexical database of English words that includes all nouns, verbs, adjectives, and adverbs and represents relations among each type [31]. For nouns, there are three main relations namely: Hyponymy, Hypernymy and Holonymy, where Hyponymy is a "type-of" relation, Hypernymy is a "is a" relation, and Holonymy is "part of" relation between two nouns. Figure 3.2 shows a partial WordNet structure for the "Computer Science" noun in which "Information Science", "Information Processing" and "Informatics" are Holonym of the Computer Science. Also "Engineering" is a Hypernym and "Computing Object" and "Artificial Intelligence" are Hyponyms.

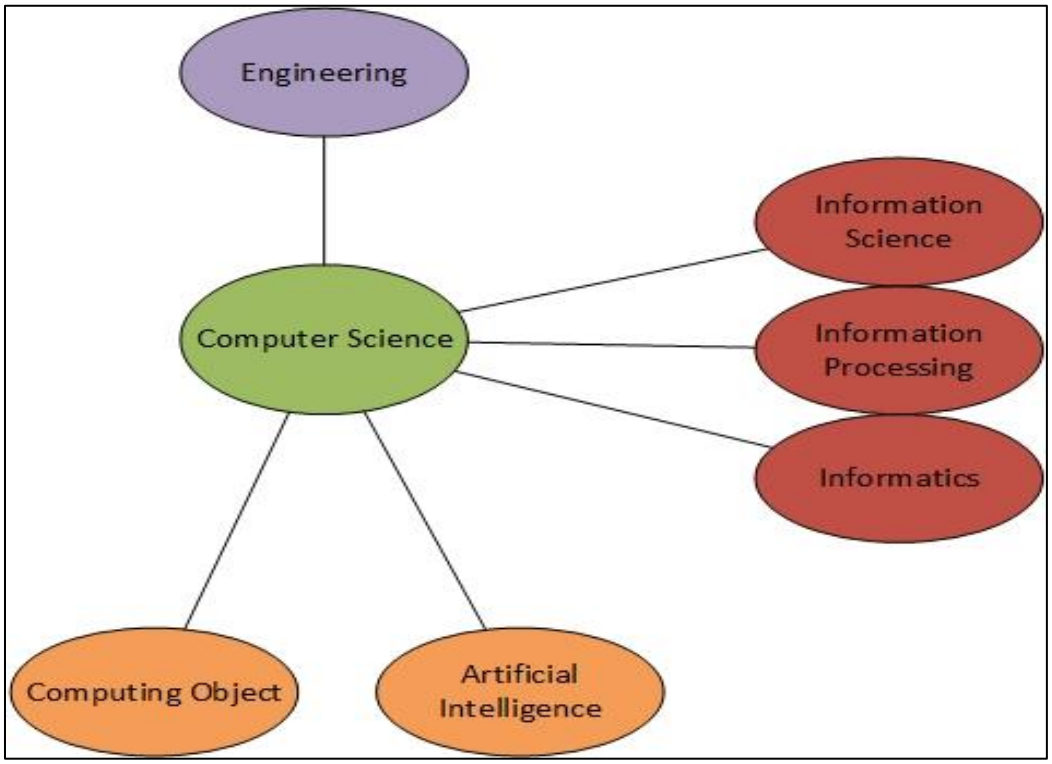

Figure 3.2 Partial structure of WordNet

3.3. Modeling User Interests

In the field of human-computer interaction, a user model is defined to be a structure that represents the user's interests for providing personalized services [32, 33, 34]. In this thesis we present a user model named *Profile Model* which is a type of domain ontology model that captures the semantics of interests in a domain. The Profile Model is a hierarchical structure that contains a set of concepts representing user long-term interests and their relations in a domain.

Ontology is a model of entities or concepts within a domain and the relationships among them. Consider the sample domain ontology provided in Figure 3.3 which visualizes some concepts and their relations in the computer domain. "Computers" is the root of the hierarchy which has "Hardware," "Software," and "Security" as its sub-concepts. In other words, Hardware, Software and Security are related to Computer conceptually; however, the relation's strength between these concepts and Computer are not the same. Software is strongly related to Computers in comparison to Security, because there are many other concepts that are super-concepts of Security such as Home, Military, and etc. Also in Figure 3.3, the relation between Programming and Software is stronger than the relation between Hardware and Programming. This is because Software is developed by programming but Hardware merely requires small codes to add complex functionalities.

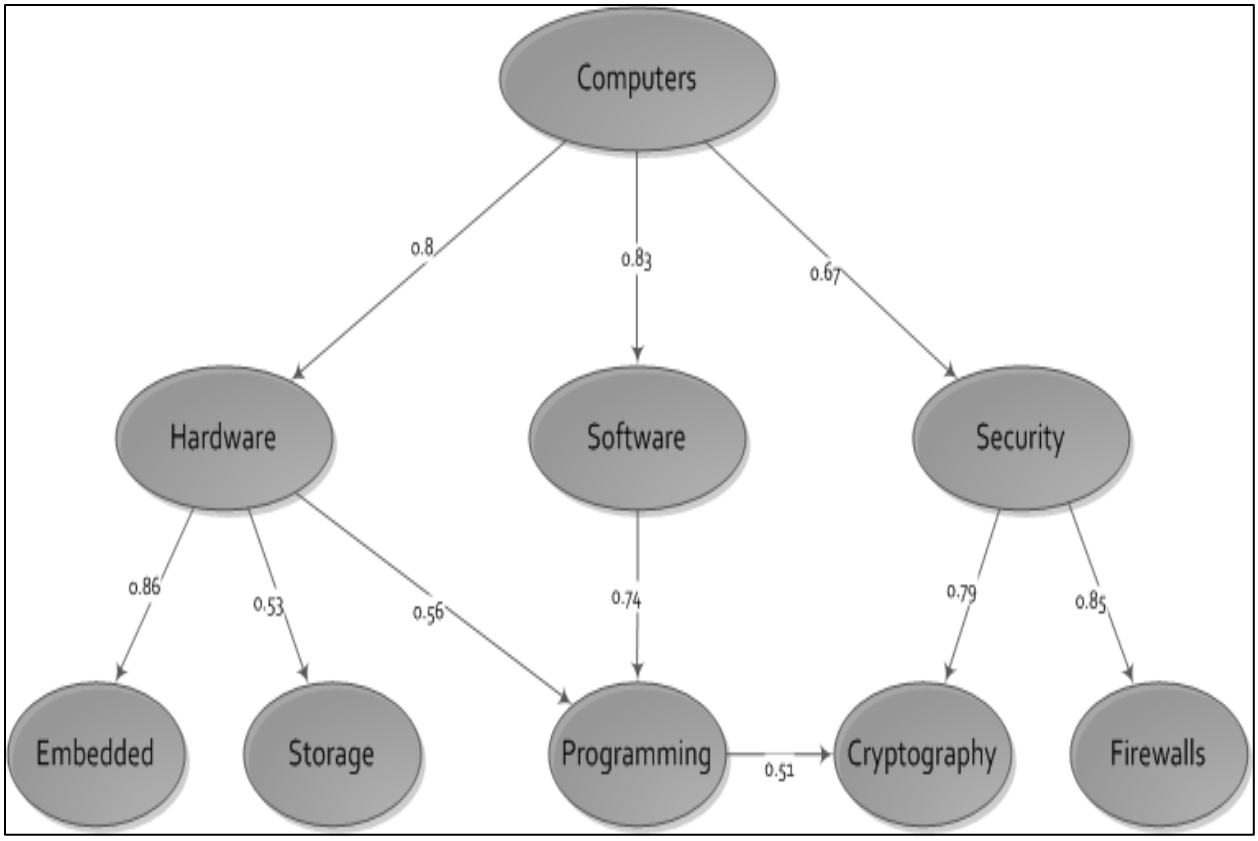

Figure 3.3 A sample of Domain Ontology

In the field of Cognitive Science, Semantic Network is one of the most accepted methods in which to represent the process of storing knowledge in long-term memory [20]. A Semantic Network is a directed graph in which each node contains a concept and edges represent relations among these nodes. Each edge has its own label that determines the type of relation between two concepts. A person's knowledge is a set of concepts that are related to each other; therefore the Semantic Network is a useful approach to represent knowledge. The neighbors and their relationship with a concept help to clearly define the meaning of this concept.

In Semantic Networks, one of the methods used to process information is Spreading Activation Theory [21]. This method helps to propagate information through the network by spreading the value of each active node to its neighbors. The process starts by activating some

nodes in the network and then iteratively updates the value of all neighbors of active nodes. Given an activate node x and a destination node y, the value of node y after applying the Spreading Activation Theory is calculated based upon the Eq.3.1.

$$V_{y,t+1} = V_{y,t} + V_{x,t}' \; w_{x,y}' \; (1-a) \qquad a \in [0,1) \tag{3.1}$$

Where V represents the gained value by node y in time $t+1$; $w_{x,y}$ represent the weight of relation between node x and node y and α is a decay factor that represents energy loss during the spreading process.

In general, the input of the process is a list of activated nodes associated with their initial values; thus the gained value of each non-activated node is defines as in Eq.3.2.

$$V_{y,t+1} = V_{y,t} + \sum_{x \in I} V_{x,t} \times \prod_{e_{j,k} \in P_{y,x}} W_{j,k} \times (1-\alpha)^{|P_{y,x}|} \tag{3.2}$$

Where $I = \{i_1, i_2, \ldots, i_s\}$ represents the list of activated nodes, $P_{y,x}$ represents the shortest path between node y and node x, $e_{j,k}$ represents an existing edge in a given path and $|P_{y,x}|$ represents the length of the shortest path between node x and node y.

Figure 3.4 is an example of spreading activation theory. In Figure 3.4 (a), "Cryptography" is activated with value 1 and the energy loss is 0. In the first iteration as shown in Figure 3.4 (b) the gained values of "Security" and "Programming" are calculated to be 0.79 and 0.51 respectively. For the next iteration as shown in Figure 3.4 (c), "Security" and "Programming" may be assumed to be active nodes with calculated values. After that, the gained value of "Software,", "Hardware," and "Computers" will be calculated.

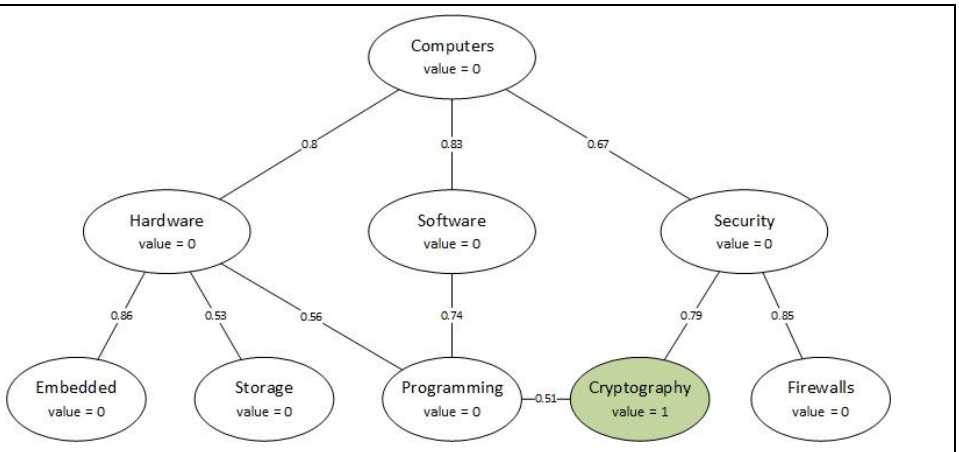

Figure 3.1(a) First iteration

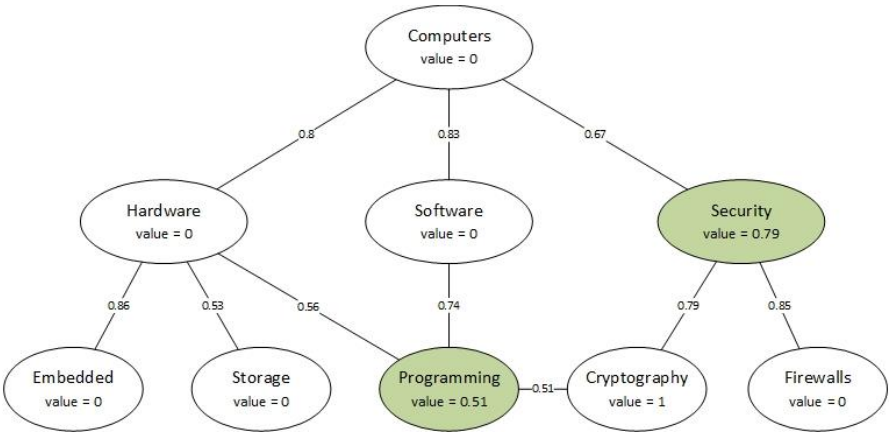

Figure 3.4(b) Second iteration

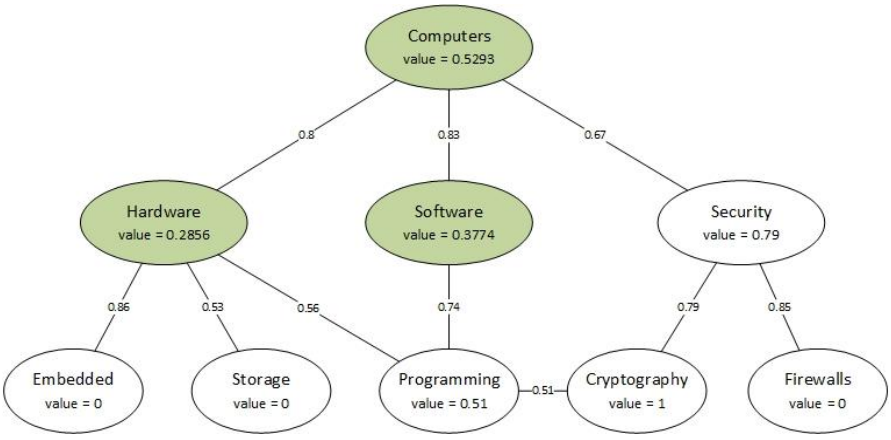

Figure 3.4(c) Third iteration

Figure 3.4 A sample of spreading activation theory

In this study, because Profile Model is structurally close to Semantic Networks, the Spreading Activation Theory is employed to update the value of each long-term interest in this model.

3.4. Ranking System

The ultimate goal of UInterestManager is to answer the question of what might be the next user's interest. Because the system is unable to predict the next interest with 100% accuracy, the UInterestManager generates a ranked list of top K interests for each user. Conversely, the next user's interest may be biased by specific situations such as sudden decisions or the user's thoughts, so predicting only one interest is not really useful and accurate.

Our approach to create a ranked list is through using a rank function that utilizes all information about user interests including Lamport timestamp [30] which is a logical time to determine the order of user interests over the time. The rank function estimates the interestingness of each preference by using the current popularity score and the last time that the interest was recalled. The Learning to Rank techniques are employed to design and train the rank function.

CHAPTER 4: UInterestManager System

4.1. Interest Extractor

To capture interests from a text document, a word segmentation algorithm was employed to split the document into words and stop words were removed by using WordNet and JWNL (Java WordNet Library). To represent the remaining part of the document, a Bag of Terms model [22] was used in the Vector Space model [23]. Therefore, the document d_i is represented by a vector as shown in Eq.4.1.

$$\vec{d}_i = < w_1, w_2, \ldots, w_n > \tag{4.1}$$

Where w_i represent different words and n is the total number of words in the document after eliminating all stop words. In the next step, the frequency of each word in w_i is calculated, see Eq.4.2.

$$d_i = \left(w_1^{k_1}, w_2^{k_2}, \ldots, w_n^{k_n} \right) \tag{4.2}$$

Where $w_i^{k_i}$ are the words with frequency k_i. JWNL is then used to determine all terms and update the frequencies. Also, by using Porter Stemming [25], terms and words are reduced to their stem. The next step involves merging all synonyms by employing JWNL and synonyms set in WordNet. The new set of words and terms are represented in Eq.4.3.

$$d_i = \left(t_1^{l_1}, t_2^{l_2}, \ldots, t_m^{l_m} \right) \tag{4.3}$$

Where, t_i are terms or words and l_i show their frequencies. To reduce the effect of big texts that may influence the user interests, Normalized Term Frequency method [24] was employed in order to normalize the frequencies, see Eq.4.4.

$$f_i = \frac{l_i}{\sum_{j=1}^{m} l_j} \tag{4.4}$$

Where f_i is a list of numerical values between 0 and 1 inclusively that represents the normalized frequency of term t_i. Finally, top p frequent terms are selected to represent the document d_i. The number p is a threshold that must be chosen wisely (based upon experience) in order to avoid affecting the performance of the system. The final term vector for a given document d_i is represented in Eq.4.5.

$$\overrightarrow{d_i} = \left(t_1^{f_1}, t_2^{f_2}, \ldots, t_p^{f_p}\right) \tag{4.5}$$

4.2. Profile Model

The Profile Model is the core component of UInterestManager that is used to manage the user's long-term interests. There are two approaches used to construct Profile Model: 1) Using a dynamic approach to build the Profile Model during the lifetime of the system. This approach is very inaccurate since there are too many factors that may affect the structure such as using UInterestManager just for a specific purpose which results in an incomplete profile. 2) Use a pre-built profile with all concepts and then update the model based upon user behaviors, documents and interests. This solution helps to provide a complete profile for a user with low overhead for

updating interests. For managing long-term interests in UInterestManager, the second approach was chosen.

To construct a profile model we adopted the modified version of Sieg *at el* [17], to build a domain ontology using Open Directory Project which organizes web pages conceptually in a hierarchical structure. For each topic, some of the related web pages were selected to create a vector of keywords that represent the concept. For concept C_i a term vector $\overrightarrow{Vc_i}$ was created by using a process that is shown in Table 4.1.

I. For all web pages that belong to C_i or its sub concepts:

i. Extract textual information from the web page.

ii. Create a dictionary of all terms and words by using WordNet and JWNL.

iii. Remove all high frequency but not-related semantically from the dictionary.

iv. Reduce all words to their stem by utilizing Porter Stemming algorithm.

v. Compute the frequency of each term and word in the dictionary.

vi. Merging all synonyms in the dictionary and update the frequencies. So the result is represented as in Eq.4.6.

$$\overrightarrow{p_i} = \left\{ t_{i,1}^{l_1}, t_{i,2}^{l_2}, \ldots, t_{i,s}^{l_s} \right\} \tag{4.6}$$

vii. Calculate the weight of each term or word by using traditional tf/idf method [26] using Eq.4.7.

$$w_{i,j} = tf\left(t_{i,j}\right) \times \log \frac{N}{n_i} \tag{4.7}$$

Where $tf\left(t_{i,j}\right)$ is the term frequency of $t_{i,j}$, N is the total number of documents that

are related to the concept C_i and n_i represents the number of web pages that contain $t_{i,j}$.

II. The aggregate representation of C_i's documents, Dc_i, is a union of all documents in sub concepts of C_i and individual web pages under C_i.

III. $\overrightarrow{Vc_i}$ is calculated as in Eq.4.8 and Eq.4.9.

$$\overrightarrow{Dc_i} = \bigcup_{d \in Dc_i} \vec{d} \tag{4.8}$$

$$\overrightarrow{Vc_i} = \frac{\overrightarrow{Dc_i}}{\sum_{t_{i,j} \in Dc_i} w_{i,j}} \tag{4.9}$$

Finally the computed term vector is normalized to unit term vector.

Table 4.1 Process of building the profile model

Classical cosine similarity measure [23, 27] is employed to calculate the weights of links between concept C_i and its sub concept C_j as in Eq.4.10.

$$w_{i,j} = \frac{\overrightarrow{Vc_i} \cdot \overrightarrow{Vc_j}}{\left|\overrightarrow{Vc_i}\right| \times \left|\overrightarrow{Vc_j}\right|} \tag{4.10}$$

To avoid of the effect of interests with large weights thorough the spreading activation process, all weights are recalculated to make sure the sum of all the weights is equal to 1. Figure 4.1 shows a partial Profile Model.

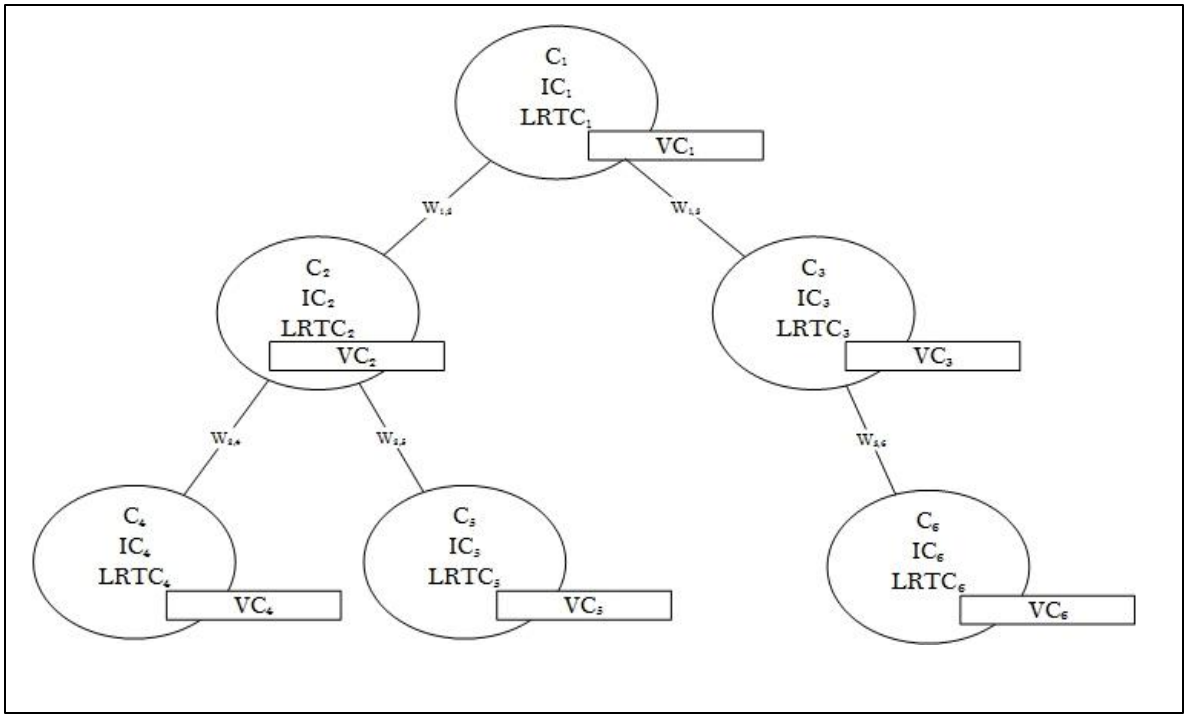

Figure 4.1 Partial Profile Model

As shown in Figure 4.1, each node contains four elements: i) a concept name C_i, ii) a value Ic_i, iii) associated term vector $\overrightarrow{Vc_i}$, and v) the last recall time $LRTc_i$ which is used to store the logical time of last recall of the concept. Building the Profile Model is a one-time process where after creating the Profile Model, the structure is not changed or updated except the values of Ic_i and $LRTc_i$. Each registered user in the system has a Profile Model that is structurally the same as other users' Profile Model but with different Ic_i and $LRTc_i$ for each concept.

We use Profile Model to manage long-term interests, and for modeling short-term interests, a queue with a fixed size length len is used. Using a queue allows the system to store ordered recent interests, which represents short-term preferences in a FIFO data structure. Each cell of the queue stores a tuple $\langle C, V \rangle$, where C is a concept and V represents the computed value of the interest for a text document.

4.3. Updating User Interests Algorithm

Table 4.2 shows the process of updating user interests after receiving a new input.

Input:

A triple of input in time t: <user id u_i, text document d_j, influence factor of the document if_{d_j} >

PMu_i: Profile Model for user u_i

Qu_i: queue of short-term interest for user u_i

SA: set of active nodes which is empty

SU: set of updated nodes which is empty

Process:

1. Compute the term vector of document d_j using the Interest Extractor system.

$$\vec{d_j} = \left(t_1^{f_1}, t_2^{f_2}, \ldots, t_p^{f_p}\right)$$

2. For all concepts C_t in PMu_i

2.1. Calculate the similarity between $\overrightarrow{Vc_t}$ and $\vec{d_j}$, using cosine similarity measure Eq.4.11.

$$sim(C_t, d_j) = \frac{\overrightarrow{Vc_t} \cdot \vec{d_j}}{\left|\overrightarrow{Vc_t}\right| \times \left|\vec{d_j}\right|} \quad (4.11)$$

2.2. If $sim(C_t, d_j) > Threshold$ then

$SA.add(C_t)$

3. For all concepts $C_k \in SA$

   3.1. Add the concept C_k to Qu_i

   $$Qu_i.enqueue(\langle C_k, sim(C_k, d_j) \times if_{d_j} \rangle)$$

   3.2. Add the concept C_k to SU

   $$SU.add(C_k)$$

   3.3. Update information of concept C_k in PMu_i using Eq.4.12 and Eq.4.13.

   $$Ic_k = Ic_k + sim(C_k, d_j) \times if_{d_j} \quad (4.12)$$

   $$LRTc_k = t \quad (4.13)$$

   3.4. For all ancestors C_m of C_k in PMu_i

      3.4.1. Update the value of C_m using spreading activation theory using Eq.4.14.

      $$Ic_m = Ic_m + \prod_{e_{x,y} \in P_{m,k}} w_{x,y} \times sim(C_k, d_j) \times if_{d_j} \times (1-\alpha)^{|P_{m,k}|} \quad (4.14)$$

      3.4.2. Add the concept C_m to SU

      $$SU.add(C_m)$$

4. For all concepts $C_p \notin SU$ update the value of the concept as shown in Eq.4.15.

   $$Ic_p = Ic_p \times \delta^{LRTc_p - t} \quad (4.15)$$

Table 4.2 Updating the model algorithm

In Table 4.2, there are two sections: input and process. The input of the system is a triple of user identity, the text document that has read by the user, and the influence factor that demonstrates the importance of the document representing the user's interests. The Profile

Model of the user and the user's queue of short-term interests are required. There are two additional data structures in which to store intermediate generated data.

The updating process starts by i) Using Interest Extractor module to create a term vector of a given document. ii) All related concepts that represent implicit interests in the document are selected. A threshold is used to ignore concepts with low correlation to the document. In other words, the system just selects concepts which accurately represent implicit interests. The value of threshold is defined based upon empirical experiences but it must be chosen carefully to avoid updating too many concepts or ignoring related concepts. Iii) The system updates all related concepts. First, all selected concepts in the last step are added to the queue of short-term interests. Then, the value and last recall time of the selected concept are updated. The UInterestManager uses a logical time system for each user that is initialized to 1 and after receiving a new input for the user it is incremented by 1. To update the value of a concept, the system adds the similarity value between the concept and the document multiplied by the influence factor to its previous value. v) All ancestors of a selected concept are updated based on Spreading Activation Theory. This step is required because concepts are related semantically so they are affected if their neighbors receive attention.

To avoid saturated value of the interests score, in the last step, the values of all concepts that are not updated based upon a given document are decreased. The system decreases the value of a not-recalled concept using the last recall time of the concept. It is based upon the simple hypothesis that if an interest is not recalled for a long period of time, the user will probably forget about it. In Eq.4.15, δ is the decay factor that is greater than 1.

4.4. Ranking Function

To generate a ranked list of interests in time $t+1$ for a user, a rank function, which utilizes long-term and short-term interests, is designed. The rank function computes the score of each interest in time $t+1$ and then the system creates a list of top K interests. The rank function is represented in Eq.4.16.

$$score_{t+1}(C_i) = \alpha \times Ic_i + \beta \times (t+1-LRTc_i) + \sigma \times f(C_i) \quad (4.16)$$

Where α and β are constant values that define the importance of current interest value and last recall time respectively. $f(C_i)$ for user u_i measures the interestingness of the concept C_i in recent time and is computed as shown in Eq.4.17.

$$f(C_i) = \sum_{\langle C_i, V_j \rangle \in Qu_i} V_j \quad (4.17)$$

Where σ is a constant value to normalize the effect of short-term interests on the interest score. To find the best values of α, β and σ the rank function must be trained. For the training process, a training data set, which is a set of 5-tuples (value of C_i in time t, last recall time of C_i, $f(C_i)$ in time t, t, position of C_i in the ranked list in time $t+1$), is required.

.

CHAPTER 5: RESULTS & EVALUATION

In this chapter, the process of testing the UInterestManager, an evaluation and an explanation of results is detailed. Because there is no benchmark or data set for evaluation of the user models' methods which depend upon textual document, we used text documents that are indexed in Open Directory Project.

5.1 The Evaluation of UInterestManager

In order to evaluate the UInterestManager methodology, an experiment was conducted using real data. First a spider was developed to build the Profile Model from the Open Directory Project website. As of February 2013, there were more than 5,000,000 web pages that have been categorized and over 1,000,000 concepts in the Open Directory Project. To create a data set for the experiment we used a branching factor and depth level of five. With these configurations, the data set contained more than 20000 documents with different concepts. Our data set was a set of tuples that contains a document that is extracted from a web page and the related concept.

To evaluate the system, the following steps were carried out: i) create the Profile Model ii) train the ranking function and iii) evaluate the performance of the system. To achieve these goals, the data set were divided into three sets including a *model training set*, a *ranking training set* and a *test set*. To create the test set, 10 concepts were selected and from each concept twelve documents were added to the test set. Then these documents were removed from the data set. The list of these concepts is shown in Table 5.1. The ranking training set was created by

selecting ten documents for each concept which was used to create the test set. These documents were added to the ranking training set and then were removed from the data set. Finally all remaining tuples in the data set were added to model training set.

Fraud	Soccer
Environmental Chemistry	Calculus
Home Automation	Public Safety
Weddings	Woodcraft
Poker	Tango

Table 5.1 List of concepts for the test set and ranking training set

To build the Profile Model, the model training set was used with the algorithm shown in Table 4.1. Then the system was developed to evaluate the UInterestManager.

In order to train the rank function accurately, RankSVM method [28], a widely accepted and highly effective ranking method, was used; the selected values of constants are shown in Table 5.2. To create a training set for the RankSVM, we used ranking training set.

Constant name	Description	Value
p	The length of the term vector for a document	30
α	Energy decay in spreading activation theory	0.96
δ	Decay factor in spreading activation theory	1.05
threshold	The minimum of similarity between a concept and a document	0.4
len	Size of queue of short-term interests	5

Table 5.2 The chosen values of constants

Following the training process, the rank function was shown in Eq.5.1 was used

$$score_{t+1}(C_i) = 1.17 \times Ic_i + 1.84 \times (t + 1 - LRTc_i) + 3.01 \times f(Ci) \quad (5.1)$$

The values of coefficients in this formula show that the short-term interests affect the score more than the long-term interests. The elapsed time for a concept is also an important factor in predicting the user's future interest.

After updating the rank function of the UInterestManager, we used the test set to evaluate the performance of the system. We created 10 tuples $\langle D, L \rangle$ where D is an ordered list of 12 documents and L is a list of interests related to D. Table 5.3 shows the process of creating these 10 tuples.

Input:

The test set $TS = \{(d_1, c_1), ..., (d_{120}, c_{120})\}$

Process:

1. For $i = 1$ to 10
    1.1. Select 6 documents with the same concept from the TS and put them randomly in D_i
    1.2. Remove those documents from TS
2. For $i = 1$ to 10
    2.1. Select 6 documents randomly from the TS and put them randomly in D_i
    2.2. Remove those documents from TS
3. For $i = 1$ to 10
    3.1. For each document with concept C_i in D_i count the number of documents with the same concept (DS)

3.1.1. If $(DS > 1)$ then

3.1.1.1. $L_i.add(C_i)$

3.2. For document d_i in position 11 or 12 in D_i

3.2.1. $L_i.add(C_i)$

Table 5.3 The process of generating test tuples

The output of the process in Table 5.3 is a set of ten tuples $\langle D, L \rangle$ which simulates a user that uses the system for 10 times. Without loss of generality, the influence factors for all documents were considered to be 1. To perform an experiment with UInterestManager, we used the same configuration as shown in Table 5.2. We did the following steps for each tuple $\langle D, L \rangle$: i) send all documents in D in order with influence factor 1 as an input to the system. ii) Send a query to the system to get a ranked list of interests. iii) For each preference in the generated ranked list, if exists in L assign 1 otherwise 0.

In the end there were10 sorted lists with their judged values. An example of the list is shown in Table 5.4.

Order	Interest	Judged value
1	Support Vector Machines	1
2	Machine Learning	1
3	Data Mining	0
4	Data set	1
5	Programming	1
6	Artificial Intelligence	0
7	Fuzzy	0

Table 1.4 An Example of a judged list

5.2. Evaluation and Results

Normalized Discounted Cumulative Gain (NDCG) measure [29], which is a well-known and highly effective measure in Learning to Rank and information retrieval fields, was used to evaluate the performance of the system. There are two assumptions for using NDCG as a measurement in order to evaluate the performance of the UInterestManager: 1) The most probable interests should be displayed at the top of the ranked list, making it the most useful and accurate.; and 2) The most interesting concepts are more useful than marginally related preferences which are better than unrelated interests. For each ranked list, NDCG is calculated over the top 10 results (NDCG@10) to measure the performance of the system.

Jiang and Tan proposed OnToSearch [16], which models the long-term interest for a user from search queries and is considered one of the most state-of-the-art methods in managing user interests. This method was modified in order to generate a ranked list of interests in real time. Then, the NDCG@10 was calculated with the same documents and interests as used in the evaluation of the UInterestManager. The results are presented in Figure 5.1.

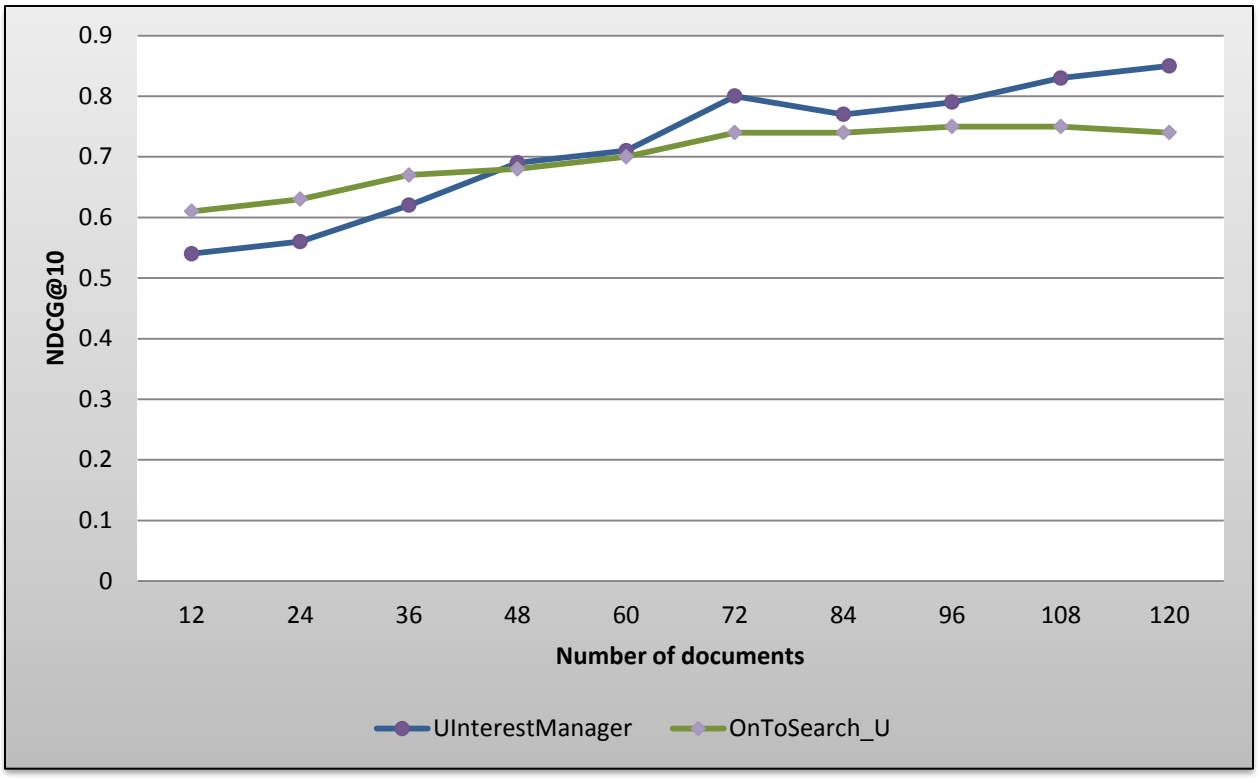

Figure 5.1 NDCG comparisons of UInterestManager and OnToSearch_U

As shown in Figure 5.1, increasing the number of documents, improves the performance of our UInterestManager system. This is because the values of interests need to be converged and the system requires more active concepts to perform in a more efficient manner. Additionally, at 84 documents the performance of the UInterestManager is degraded that is the result of lack of enough documents about new preferences previously.

The OnToSearch_U method merely manages the long-term interests for a user to predict the next interest which results in poor performance for the large number of documents in comparison to that of the UInterestManager. These results show that using short-term interests significantly increases the accuracy of predicting interests.

CHAPTER 6: CONCLUSION AND FUTURE WORKS

6.1 Conclusion

This thesis has revealed a new method for extracting user interests from textual documents and to manage user preferences. A user model was created based upon domain ontology using Open Directory Project to manage both long-term and short-term interests. Then for each new document, the users' explicit and implicit interests were extracted and the Profile Model was updated. Finally, the system generated a ranked list regarding user interests in real time. This experiment shows significant improvement as opposed to that of other state-of-the-art method.

6.2 Future Works

One possible direction for future works might be using all types of documents, such as images, videos, and search queries as a way to collect even more interests for a user and improve the accuracy of the system. Another direction might be using a dynamic user model which would manage user interests more effectively and more realistically.

REFERENCES

[1] Bilenko, M., White, R. W., Richardson, M., & Murray, G. C. (2008, July). Talking the talk vs. walking the walk: salience of information needs in querying vs. browsing. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 705-706). ACM.

[2] Carmagnola, F., Cena, F., Cortassa, O., Gena, C., & Torre, I. (2007). Towards a tag-based user model: how can user model benefit from tags?. In *User Modeling 2007* (pp. 445-449). Springer Berlin Heidelberg.

[3] Ma, Y., Zeng, Y., Ren, X., & Zhong, N. (2011). User interests modeling based on multi-source personal information fusion and semantic reasoning. In *Active Media Technology* (pp. 195-205). Springer Berlin Heidelberg.

[4] Badi, R., Bae, S., Moore, J. M., Meintanis, K., Zacchi, A., Hsieh, H., ... & Marshall, C. C. (2006, January). Recognizing user interest and document value from reading and organizing activities in document triage. In *Proceedings of the 11th international conference on Intelligent user interfaces* (pp. 218-225). ACM.

[5] Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J., Miller, B., & Riedl, J. (1998, November). Using filtering agents to improve prediction quality in the grouplens research

collaborative filtering system. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work* (pp. 345-354). ACM.

[6] Seo, Y. W., & Zhang, B. T. (2000, June). Learning user's preferences by analyzing Web-browsing behaviors. In *Proceedings of the fourth international conference on Autonomous agents* (pp. 381-387). ACM.

[7] Liang, T. P., & Lai, H. J. (2002, January). Discovering user interests from web browsing behavior: An application to internet news services. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on* (pp. 2718-2727). IEEE.

[8] Zheng, L., Cui, S., Yue, D., & Zhao, X. (2010, August). User interest modeling based on browsing behavior. In *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on* (Vol. 5, pp. V5-455). IEEE.

[9] Li, L., Yang, Z., Wang, B., & Kitsuregawa, M. (2007). Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In *Advances in Data and Web Management* (pp. 228-240). Springer Berlin Heidelberg.

[10] Morita, M., & Shinoda, Y. (1994, August). Information filtering based on user behavior analysis and best match text retrieval. In *Proceedings of the 17th annual international ACM*

SIGIR conference on Research and development in information retrieval (pp. 272-281). Springer-Verlag New York, Inc.

[11] Oard, D. W., & Kim, J. (1998, July). Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems* (pp. 81-83). Wollongong.

[12] Shen, X., Tan, B., & Zhai, C. (2005, August). Ucair: Capturing and exploiting context for personalized search. In *Proceedings of the ACM SIGIR 2005 Workshop on Information Retrieval in Context (IRiX)* (p. 45).

[13] Teevan, J., Dumais, S. T., & Horvitz, E. (2005, August). Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 449-456). ACM.

[14] Widyantoro, D. H., Ioerger, T. R., & Yen, J. (2001). Learning user interest dynamics with a three-descriptor representation. *Journal of the American Society for Information Science and Technology*, *52*(3), 212-225.

[15] Tan, A. H., & Teo, C. (1998, May). Learning user profiles for personalized information dissemination. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on* (Vol. 1, pp. 183-188). IEEE.

[16] Jiang, X., & Tan, A. H. (2009). Learning and inferencing in user ontology for personalized Semantic Web search. *Information sciences*, *179*(16), 2794-2808.

[17] Sieg, A., Mobasher, B., & Burke, R. D. (2007). Learning Ontology-Based User Profiles: A Semantic Approach to Personalized Web Search. *IEEE Intelligent Informatics Bulletin*, *8*(1), 7-18.

[18] Pretschner, A., & Gauch, S. (1999). Ontology based personalized search. In *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on* (pp. 391-398). IEEE.

[19] Xing, K., Zhang, B., Zhou, B., & Liu, Y. (2011, October). Behavior based user interests extraction algorithm. In *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing* (pp. 448-452). IEEE.

[20] Anderson, J. R. (2013). *Language, memory, and thought*. Psychology Press.

[21] Anderson, J. R. (1983). A spreading activation theory of memory. *Journal of verbal learning and verbal behavior*, *22*(3), 261-295.

[22] Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1). Cambridge: Cambridge University Press.

[23] Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613-620.

[24] Li, Y. H., & Jain, A. K. (1998). Classification of text documents. *The Computer Journal*, *41*(8), 537-546.

[25] Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, *14*(3), 130-137.

[26] Chowdhury, G. (2010). *Introduction to modern information retrieval*. Facet publishing.

[27] Salton, G., & McGill, M. J. (1986). Introduction to modern information retrieval.

[28] Joachims, T. (2002, July). Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 133-142). ACM.

[29] Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, *20*(4), 422-446.

[30] Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, *21*(7), 558-565.

[31] Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41.

[32] Fischer, G. (2001). User modeling in human–computer interaction. User modeling and user-adapted interaction, 11(1-2), 65-86.

[33] Johnson, Addie; Taatgen, Niels (2005), "User Modeling", Handbook of human factors in Web design, Lawrence Erlbaum Associates, pp. 424–439

[34] Montaner, M., López, B., & De La Rosa, J. L. (2003). A taxonomy of recommender agents on the internet. Artificial intelligence review, 19(4), 285-330.